Projet Fragrances

Classification non-supervisée de parfums

Maxime Blanchard

Table des matières

Introduction : présentation du projet Fragrances

a. Objectif

L'objectif du projet Fragrances est d'identifier des classes de parfums à partir des ingrédients utilisés dans les recettes de chacun d'eux.

Par exemple, les parfums qui partagent des notes de fruits seraient rassemblés dans la famille olfactive « fruitée », tandis que ceux où le musc est davantage présent dans la famille « musquée ».

b. Jeu de données

Pour ce faire, un jeu de données listant les ingrédients d'environ 30 000 parfums différents servira de matériau de base pour ce projet[1], les outils de de la librairie scikit-learn pour l'apprentissage de ces données et la librairie nltk pour l'analyse textuelle.

c. Méthodologie

Les familles olfactives sus-dites n'étant pas présentes dans le jeu de données, il sera tenté de les constituer à l'aide d'un algorithme d'apprentissage non-supervisé *K-means.*

D'autre part, étant donné le nombre conséquent d'ingrédients uniques différents (1415), la technique de *Natural Language Processing* (NLP) dite de *bag-of-words* a paru à la fois être intuitivement la plus adaptée et la plus abordable techniquement.

1 Accessible à cette adresse : https://www.kaggle.com/datasets/imerghichi/notes-perfumes

1. Analyse exploratoire des données

a. Caractéristiques générales

Les dimensions du jeu de données sont de 31275 lignes et 23 colonnes.

Les 23 colonnes sont :

- 'brand' : la marque du parfum ;
- 'title' : le nom du parfum ;
- 'note1' à 'note20' : chaque note est un ingrédient différent. La recette d'un parfum peut être composée jusqu'à 20 ingrédients ;
- 'gender' : genre auquel est destiné le parfum.

Concernant les valeurs uniques, il y a :

- 3123 marques différentes ;
- 31077 noms de parfums différents ;
- 1416 ingrédients différents ;
- 3 genres différents.

Toutes les variables sont catégoriques et présentent des valeurs de type *string* ou *NaN*.

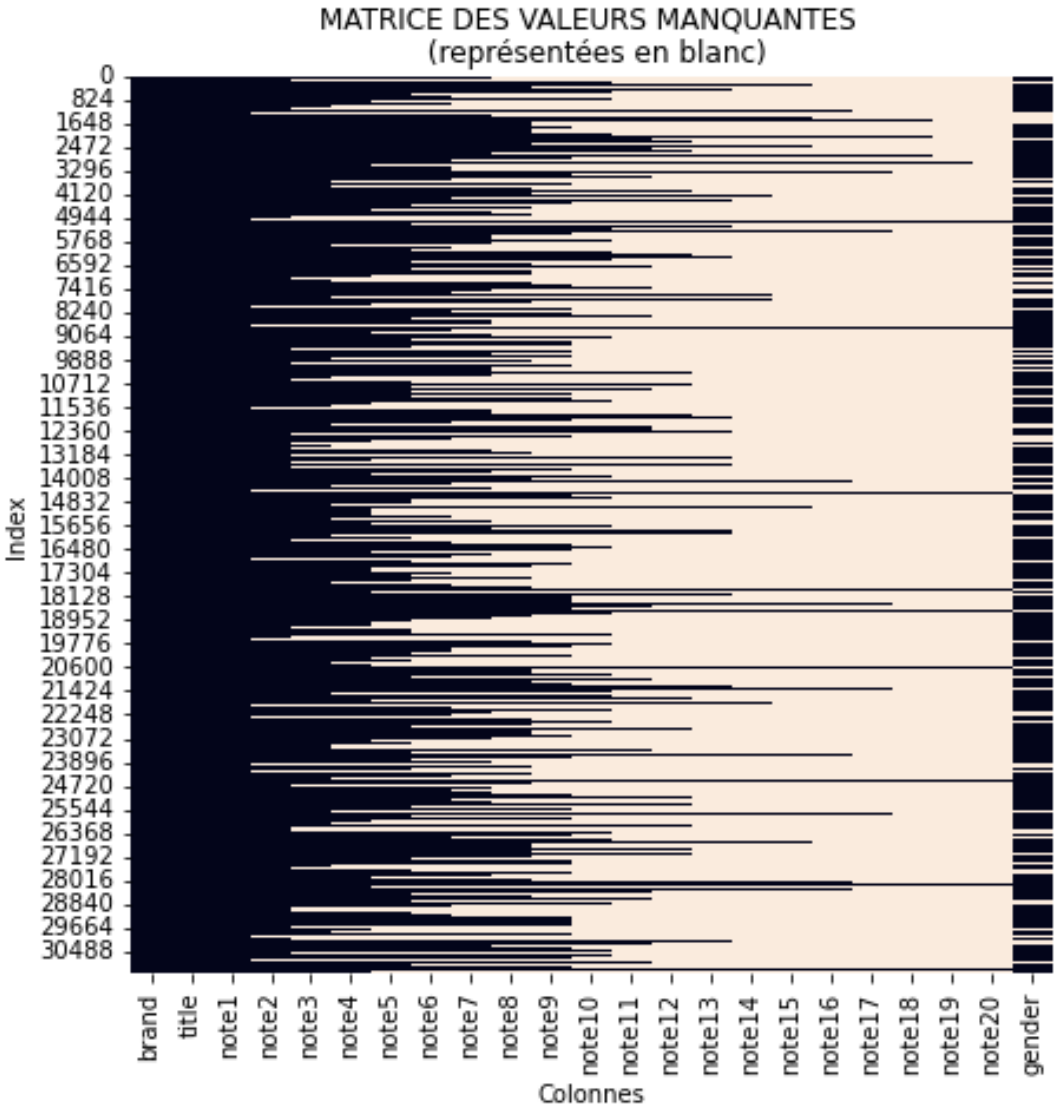
Voici un tableau récapitulatif de ce qui vient d'être dit :

23 colonnes

	Brand	**Title**	**Note1 à note20**	**gender**
31275 lignes	la marque du parfum	le nom du parfum	chaque note est un ingrédient différent. Un parfum peut être composé jusqu'à 20 ingrédients	genre auquel est destiné le parfum
unique	3123	31077	1416	3
dtype	Obj	Obj	Obj	Obj

b. Distribution des valeurs manquantes

La distribution des valeurs manquantes est représentée dans la matrice suivante :



On remarque que tous les parfums ne sont pas composés du même nombre d'ingrédients, et d'autre part que la colonne 'gender' possède un nombre conséquent de valeurs manquantes.

Le tableau suivant résume plus avec plus de détail la distribution des valeurs manquantes. La ligne *Count* du tableau est le nombre de valeurs présentes dans la colonne, d'où est déduit le pourcentage de valeurs manquantes (*NA %*) *:*

Tableau récapitulatif des valeurs manquantes par colonne

Colonne	**Brand**	**Title**	**Gender**
Count	31275	31275	24802
NA %	0	0	20.7

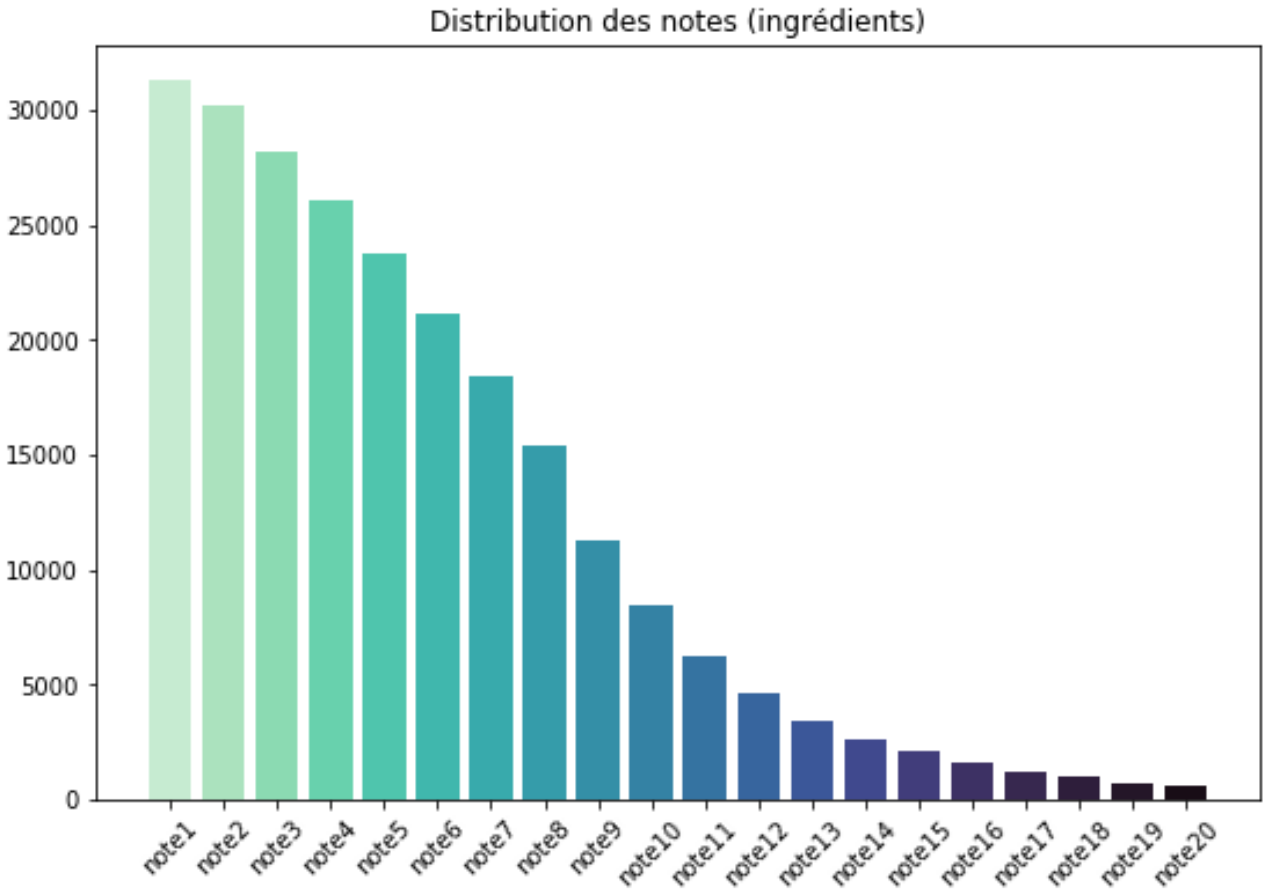
Colonne	**Note1**	**Note2**	**Note3**	**Note4**	**Note5**	**Note6**	**Note7**	**Note8**	**Note9**	**Note10**
Count	31264	30204	28156	26042	23726	21132	18430	15344	11224	8400
NA %	0.04	3.42	9.97	16.73	24.14	32.43	41.07	50.94	64.11	73.14

Colonne	**Note11**	**Note12**	**Note13**	**Note14**	**Note15**	**Note16**	**Note17**	**Note18**	**Note19**	**Note20**
Count	6278	4619	3454	2622	2056	1582	1224	952	742	581
NA %	79.93	85.23	88.96	91.62	93.43	94.94	96.09	96.96	97.63	98.14

c. Distribution détaillée des colonnes

- note1 à note20

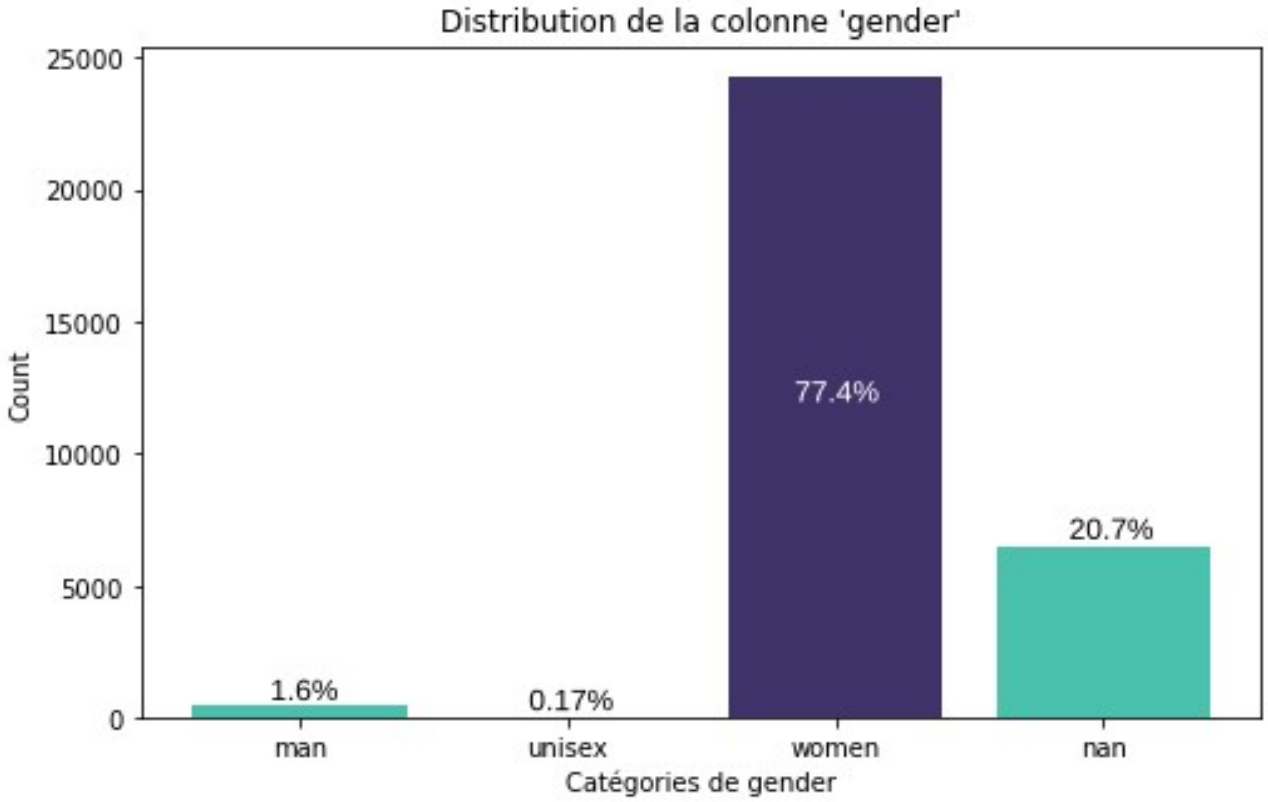
Voici le diagramme à barres de la distribution des colonnes ‘note1’ à ‘note20’ :



Le nombre d’ingrédients dans une recette suit une distribution normale décroissante. On observe qu’environ 50% des recettes ont 8 ingrédients ou moins.

- gender

Étant donné que la colonne ‘gender’ pourrait aider à catégoriser les différents types de parfums, voici représentée sa distribution détaillée :



Elle prend trois valeurs différentes : man, unisex et women (et environ 20 % de valeurs nulles). Mais cette colonne est très uniforme : la valeur women représente l'écrasante majorité des valeurs non-nulles de cette colonne. Elle est donc très uniforme, et, par conséquent, peu informative.

2. Pré-traitement Machine Learning

a. Suppression des colonnes inutiles

Les colonnes 'brand' et 'title' sont supprimées car elles ne fournissent pas d'information sur la nature intrinsèque des parfums, seulement sur leur identité. La colonne 'gender' est supprimée car, comme dit précédemment, elle est trop uniforme et peu utile d'un point de vue informationnel.

Seules les colonnes' note1' à 'note20' sont donc conservées pour le Machine Learning.

b. Standardisation des données

Pour préparer la technique de *bag-of-words* des données textuelles, on standardise ces dernières à l'aide de plusieurs fonctions regex :

- remplacement des valeurs manquantes par un espace ;
- mettre en caractères minuscules ;
- suppression des caractères spéciaux ;
- suppression des caractères numériques ;
- suppression des espaces.

Remarquons qu'on supprime les espaces pour s'assurer que les ingrédients désignés par plusieurs mots soient considérés comme un seul et même ingrédient. Par exemple :

1Bulgarian Rose™ → bulgarianrose

Grâce à ce traitement, le nombre d'ingrédients uniques a légèrement diminué à 1373.

Enfin, on concatène les ingrédients de chaque parfum, séparés par un espace, dans une nouvelle colonne 'recette'.

3. Machine Learning : classification non-supervisée K-means

a. Choix du Bag-of-words

Les valeurs textuelles ne peuvent pas être traitées directement par les technologies de Machine Learning, il faut les transformer numériquement en perdant le moins possible d'information.

Plusieurs solutions ont été envisagées : One Hot Encoding, Ordinal Encoding et bag-of-words. Les deux premières ont été écartées :

- le one hot encoding génère trop de colonnes et ne parvient pas à saisir les rapprochements possibles entre deux colonnes différentes ;
- l'ordinal encoding insère une ordinalité arbitraire qui fausse le modèle et nous arrivons à des résultats médiocres (il en résulte notamment que le K-means classe les parfums davantage en fonction du nombre de valeurs manquantes plutôt qu'en fonction des ingrédients de la recette).

Reste le bag-of-words qui offre des résultats plus concluants.

Pour utiliser le vocabulaire technique du bag-of-words :

- l'ensemble des 31 275 recettes de parfums constituent le « *corpus* » ;
- chaque recette est un « *texte* » ;
- chaque ingrédient est un « *mot* » ;
- l'ensemble des mots uniques du corpus forme le « *vocabulaire* ».

b. Encodage des données

Grâce à la librairie scikit-learn, on instancie un objet CountVectorizer qui :

- crée un *vocabulaire* (type dictionnaire) en associant chaque *mot* unique (clé) du *corpus* à une valeur unique ;
- crée pour chaque *texte* un vecteur de la longueur du *vocabulaire* (1382 éléments), dont chaque élément peut prendre la valeur 0 ou 1 qui représente si le *mot* correspondant est présent ou non dans le *texte*. L'index de l'élément correspond à la valeur du *mot* dans le *vocabulaire*.

Désormais, chaque parfum voit sa recette remplacée par un vecteur numérique et peut être traité par l'algorithme K-means.

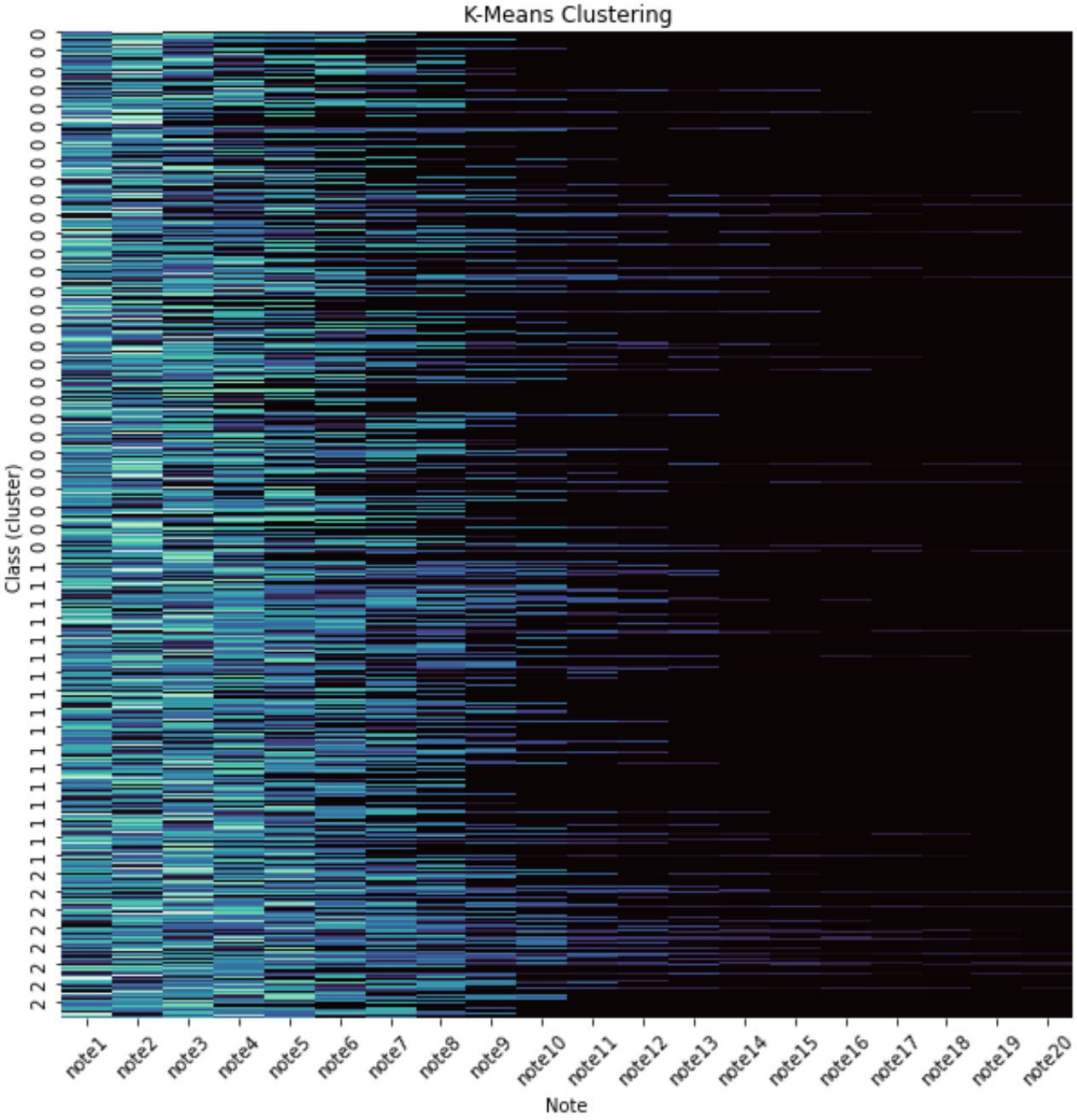
c. Classification non-supervisée K-means

L'algorithme K-means va attribuer des classes à l'ensemble des parfums selon la similarité de leurs vecteurs bag-of-words.

On commence par défaut avec trois classes (clusters) :

```
kmeans = KMeans(n_clusters=3, random_state=0,
n_init='auto')
```

Représentation graphique de la classification dans une heatmap :



En allant regarder les valeurs elles-mêmes, on constate après un rapide coup d'oeil :

- la catégorie 0 possède souvent les ingrédients rose et strawberry ;
- la catégorie 1 possède souvent les ingrédients musk et sandalwood ;
- la catégorie 2 possède souvent les ingrédients ylang-ylang et lily-of-the-valley.

De façon plus rigoureuse, on peut se servir de la librairie NLTK (Natural Language Tool Kit) pour obtenir davantage d'information.

d. Analyse textuelle avec NLTK

En utilisant la méthode .similar(), on peut savoir quels mots sont le plus souvent présents ensemble. Par exemple pour le mot 'musk' :

```
import nltk
from nltk.corpus import PlaintextCorpusReader

corpus_root = "maxbld/fragrances"
files = ".*\.txt"

corpus0 = PlaintextCorpusReader(corpus_root, files)
corpus  = nltk.Text(corpus0.words())

corpus.similar('musk')
```

On obtient :

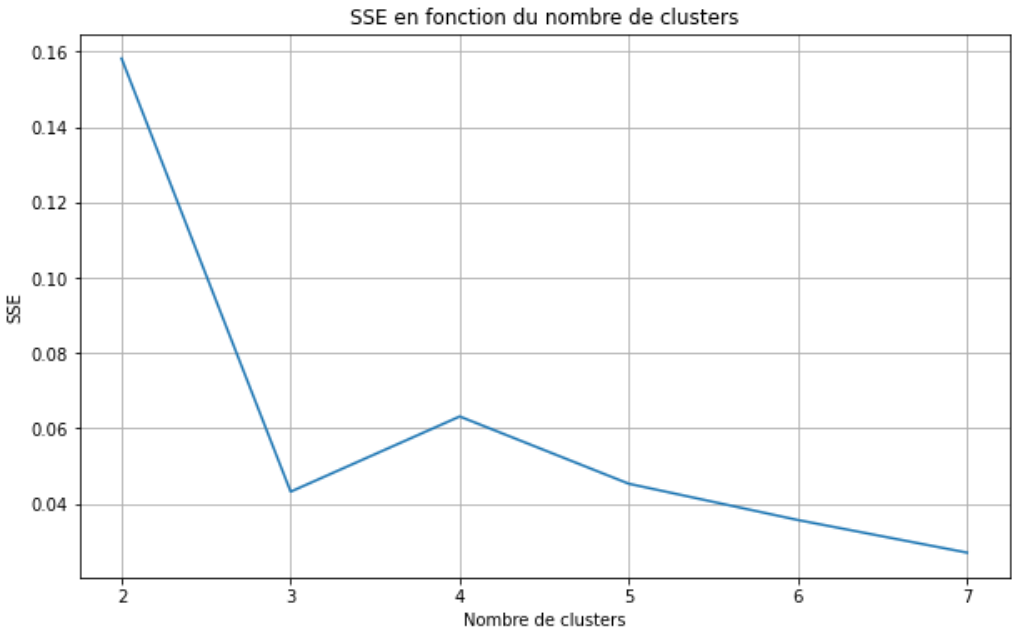
```
[out] : amber sandalwood vanilla patchouli cedar vetiver
jasmine rose
tonkabean bergamot woodynotes oakmoss whitemusk leather
mandarinorange
agarwoodoud violet iris incense benzoin
```

e. Optimisation des hyper-paramètres

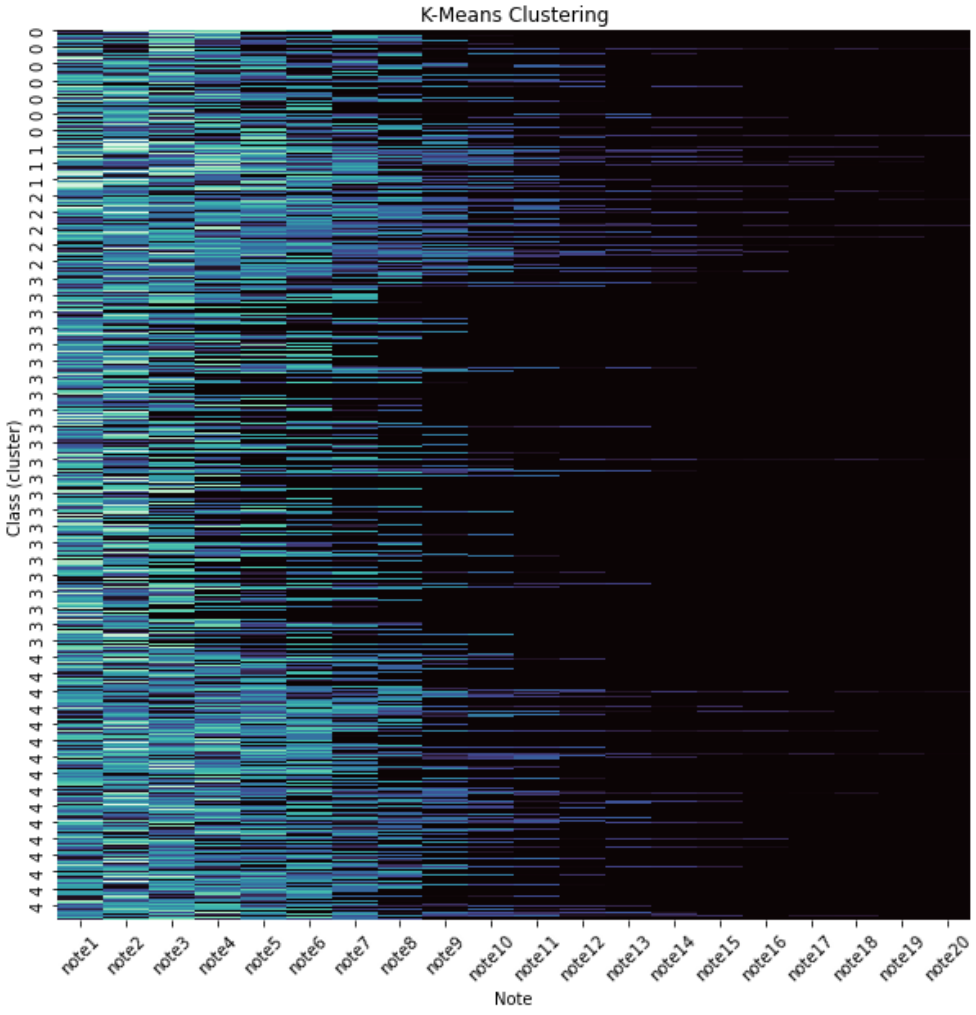
Par manque de connaissance-métier, on ne connaît pas le nombre de classes à rechercher.

La méthode Elbow peut nous aider à déterminer le nombre optimal de clusters pour K-means qui est situé à l' « elbow point »[2] de sa courbe SSE (Sum of Square Errors), c'est à dire lorsque la courbe commence à diminuer linéairement. Dans notre cas, la courbe SSE commence à diminuer de façon linéaire à partir de 5 clusters (mais 3 clusters a un meilleur score) :

2 https://vitalflux.com/k-means-elbow-point-method-sse-inertia-plot-python/



Voici donc la heatmap obtenue avec 5 clusters :



Ce résultat mériterait une analyse plus poussée, mais, à première vue, les valeurs nulles prennent trop d'importance dans la classification.

Nous finirons cette étude en exportant deux fichiers contenant les 23 colonnes de départ et la colonne label ajoutée grâce à l'algorithme K-means :

- perfumes_3clusters.csv
- perfumes_5clusters.csv

Conclusion

Bien qu'il soit difficile de juger de la pertinence des résultats basés seulement sur une similarité lexicale, il sont néanmoins beaucoup plus satisfaisants que ceux obtenus lors d'essais précédents (non-détaillés dans ce document), comme ceux à partir d'un Ordinal Encoding (cf. section 3.a.). Le modèle nous propose plusieurs classes qui semblent bel et bien proches lexicalement parlant, ce qui était le critère proposé. Cette similarité a été en partie vérifiée grâce aux outils de nltk, mais mériterait d'être davantage illustrée (et critiquée).

Concernant les améliorations possibles de ce projet, la réduction de dimensionalité n'a pas eu l'attention qu'elle méritait, faute de temps. Par exemple, en supprimant les colonnes où il y a un nombre trop élevé de valeurs manquantes, ou encore étudier la technique de *Principal Component Analysis*.

De plus, des techniques NLP plus poussées attendent encore être explorées, et les représentations en graphes de la librairie networkx semblent être une autre piste prometteuse.

Enfin, l'apport d'un expert du domaine pourrait nous aider à juger de la pertinence du clustering et à améliorer le jeu de données de départ en lui ajoutant des niveaux de granularité.